CARNEGIE-MELLON UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

SPICE PROJECT

Introduction to the Spice Users' Manual

Edited by Mario R. Barbacci

9 February 1984

Spice Document

Keywords and index categories:

Machine-readable file: [Spice]/usr/spicedoc/manual/spiceuser/overview/Spintro.mss

Supported by the Defense Advanced Research Projects Agency, Department of Defense, ARPA Order 3597, monitored by the Air Force Avionics Laboratory under contract F33615-78-C-1551. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

Table of Contents

1. Introduction and Organization of the User's Manual 3
2. Overview of Spice 4
    2.1. Server Processes 5
    2.2. Applications 7
    2.3. A Glimpse at the Future 8
3. Getting Started - The Hardware 8
    3.1. The Perq 8
    3.2. The Tablet and Pointing Device 9
    3.3. Booting Accent 10
    3.4. The Display 12
4. Getting Started - The Software 12
    4.1. Devices, Partitions, and Directories 12
    4.2. PathNames 13
        4.2.1. File Names and Extensions 13
        4.2.2. Logical Names 14
        4.2.3. Search Lists 15
        4.2.4. Absolute PathNames 15
        4.2.5. Network PathNames 16
        4.2.6. Relative PathNames 16
        4.2.7. PathName Restrictions 17
        4.2.8. Wildcards 17
5. Getting Started - The Shell 18
    5.1. Command Line Conventions 18
    5.2. Special Command Lines 20
    5.3. Using the Spice Shell 20
    5.4. The Intra-line Editor 21
        5.4.1. Command Line Editing Functions 21
        5.4.2. Automatic FileName Completion 22
        5.4.3. Retrieving Previous Command Lines 23
        5.4.4. Scroll Control a.k.a. "more-mode" 23
    5.5. Session Transcript Buffers 24
    5.6. Creating Additional Windows 24
    5.7. Initial Command Files and User Profiles. 25
    5.8. Ending a session 25
6. Basic Commands and Application Programs 25
    6.1. On-line Help 25
    6.2. File and Environment Manipulation Commands 26
        6.2.1. Directory 26
        6.2.2. Path 26
        6.2.3. MakeDirectory 27
        6.2.4. Copy 27
        6.2.5. Rename 27
        6.2.6. Delete 27
        6.2.7. Show 28
        6.2.8. Define 28

6.2.9. Alias 28
6.2.10. SetSearch 29
6.3. Producing Documents 29
6.3.1. Oil 29
6.3.2. Mint 29
6.3.3. DP 29
6.3.4. Dover 30
6.4. Using the Ethernet 30
6.4.1. Chat 30
6.4.2. Cmuftp 30
6.4.3. Update 30
6.4.4. Mercury 31
7. What To Do When Something Goes Wrong? 31
7.1. Reporting Problems 31
7.1.1. Failure to Boot 31
7.1.2. Accent Crashes 31
7.1.3. Failure to Run 32
7.2. Getting Out of the Kraut Debugger 32
7.3. Partition and Scavenger 33
7.4. When Things Look Hopeless 34
Appendix A. SpiceTalk Glossary 35
Appendix B. Perq Hardware Configurations 39
Appendix C. Booting Accent 40
Appendix D. Notation Conventions 43
Appendix E. Filename Extensions 44
Appendix F. Predefined File Names 47
Appendix G. Son et Lumiere 48
Appendix H. Key Translation and the Intra-line Editor 49
Appendix I. Spice Project Contributors 52

Problem reports

All reports of problems related to Spice or its documentation should be mailed to the ArpaNet address **Spice@CMU-CS-Spice**. This address may be abbreviated to **Spice@Spice** on CMU Computer Science Department computers. To keep track of the latest changes in systems, documentation, and procedures, all Spice users should read the "Spice" bulletin board on any of the local host-machines.

Where to find things

All Spice software is currently stored on the CMU-CS-CFS Host (a Vax Unix) in a directory tree rooted at **/usr/spice**. To get an up-to-date Spice system, including both the Accent kernel, the servers, the standard commands and utility programs, use the **Update** program, type

```
path >Sys>Spice
```

This places you in the proper directory. Now type one of the following, depending on your PERQ's model:

update Perq1
update Perq1A
update Perq2

To get just one particular system, such as **DP**, you would type

update dp

Preface to the First Edition

This Introduction to the Spice User's Manual is based on a preliminary version written by Roger Dannenberg, Kathleen Ferraro, and Richard Gumpertz. Additional material for this first edition was provided by Bernd Bruegge and Mary Thompson. Robert Baron, Thomas Newton, and David Nason provided many comments and suggestions for improving this document.

Parts of the preliminary version were extracted from previous documents written by members of the Spice project or its industrial affiliate, Perq Systems Corporation. In this regard, we wish to thank: J. Eugene Ball, Miles Barel, Jeffrey L. Eppinger, Diana Connan Forgy, Brad A. Myers, David A. Nichols, Robert D. Sansom, Donald A. Scelza, Joyce H. Swaney, and Jon A. Webb

In addition, we thank the entire Spice group for describing things to us and helping us track down details. The present and past members of the Spice project are listed in Appendix I.

The intended audience for this Introductory User's Manual varies greatly in their knowledge about computers and computer usage. Some readers will be experienced programmers, others will be experienced users of computers without ever having to write programs, and finally, there will be readers for whom Spice is their first exposure to state-of-the-art computing. We make the assumption however, that the readers are familiar with, or can easily learn about the computing facilities of the CMU-CS Department: this document will not tell you what is a"Dover", or what does it mean to "send mail to Spicc@Spice". These are items that belong in a more general CMU-CS Facilities Manual (there is one and it is updated periodically.)

To help the less experienced readers, explanations and details that are of interest only to programmers or advanced users are presented in paragraphs with wider margins and in a smaller font. These can be safely skipped on a first (or Nth) reading. In addition, a glossary of commonly used "SpiceTalk" terms is given in Appendix A.

MRB, December 1983

Acknowledgments

Spice is a cooperative effort of many individuals and groups within the Computer Science Department. Funding for the project has come from the Defense Advanced Research Projects Agency and from industry. Past and present Industrial Affiliates of Spice are Digital Equipment Corporation, International Computers Limited, Olivetti Corporation, Science and Engineering Research Council (UK), Siemens Corporation, Symbolics, Inc., Perq Systems Corporation, and Xerox Corporation.

1. Introduction and Organization of the User's Manual

Spice stands for Scientific Personal Integrated Computing Environment. It is a project within the Computer Science Department of Carnegie-Mellon University with the goal of providing the software for a computing environment based on powerful personal computers. The environment that results will be able to replace the current time-shared systems for the majority of the computing needs of the department, and it is expected that it will serve these needs into the 1990's. The project is now making its first public release of software. This software executes on Perq Systems Corporation PERQ computers, which are being used as interim development machines.

This first public release represents significant steps toward our goal. It contains major improvements relative to earlier versions. In particular, there are advancements in terms of performance and coherence, as well as new tools, such as Mint and Sapphire. Although new tools or improved versions of older tools will continue to be developed, the success of Spice depends largely on the community using these tools with sympathy but also with critical appraisal of their performance.

The documentation for this version of Spice appears in two parts. The first, the *Spice User's Manual* describes Spice for new or casual users of the system. The second, *The Spice Programmers' Manual*, provides detailed information of interest primarily to programmers.

The *Spice User's Manual* is also divided into two parts: the first part, this *Introduction to the Spice User's Manual*, provides general information on Spice and its applications. The second part of the *Spice User's Manual* is a collection of manuals describing individual components of the system.

After the Introduction and Overview of Spice, you will find three chapters describing the PERQ hardware, the file system naming conventions, and the shell. They contain instructions on how to get started and document many of the quirks and miscellaneous details that are usually discovered the hard way. The next chapter contains brief descriptions of several commands and subsystems likely to be used by most members of the CMU CS community. These include, for example, document production facilities. Following, you will find a chapter describing problems that you might encounter, together with some suggested recovery procedures. The appendices at the end of the document contain information that you will find useful at different stages of your familiarization with the system. Some of these you might want to read right away (e.g. the Glossary in Appendix A), others you could ignore until after you have had some practice sessions with the system.

The second and larger part of this Spice User's Manual consists of manuals, some of which were prepared specially for this release and some of which were already in existence. The manuals are appended to the first

part in the following order[1]:

Users Guide to the Sapphire Window Manager
Oil: The Spice Ascii Editor
*Mint: The Spice Document Preparation System**
DP: The Spice Drawing Program
Cousin Manual
*Mercury Users' Manual**
*Spoonix Manual**
*Update: A Version Control/File Transfer Facility**
Spice Commands and Utilities

The *Spice Programmers' Manual* contains the following documents:

Systems Programmers' Manual
Perq Pascal Extensions
Pascal Library
Accent Public Module Index
Matchmaker: A Remote Procedural Call Generator
Procedural Interface to the Sapphire Window Manager
User Manual for Kraut: The Spice Interim Debugger
Sesame: The Spice File System
The Preliminary Environment Manager
Accent Kernel Interface Manual
Perq Q-Code Reference Manual for Accent
Mint Reference Manual
*C/Spoonix**
The Assembler and Related Programs

2. Overview of Spice

The kernel of the Spice operating system is called Accent. For historical reasons, the whole Spice operating system has come to be called "the Accent System," or simply "Accent." Thus, the name Accent refers to both the kernel and the operating system built around it. Incidentally, there is another operating system installed on your PERQ called POS. At CMU, POS is used as a back-up system.

In many computer systems, the term "operating system" stands for some rather large, monolithic program which does most of the work. It reads and interpretes command lines typed by the users, it manages and protects files and directories, it provides network access facilities, etc. In Spice, the Accent kernel provides processes, address spaces, and message-passing primitives. Other services commonly thought of as being part of the "operating system" such as the file system, command interpreter (shell), network communication and

[1]Manual titles followed by an asterisk (*) are not included in this documentation release but will be distributed as soon as they are available.

screen management are provided by separate "server" processes which look like ordinary application or user programs.

Accent is written in a combination of micro-code and Pascal. Unless you are an advanced user, with special needs, you will rarely come in touch directly with Accent. Rather, you (and your programs) will interact with the servers built on top of the kernel. The server processes get the job done by executing "kernel calls" on your behalf.

> An Accent process is associated with a uniform virtual address space of 2^{31} 16-bit words. This address space cannot be shared with another process, and even messages are passed by copying (although there are some implementation tricks to make this efficient). As a result, a process is protected against corruption by other processes, thus preventing bugs in one process from mysteriously modifying the behavior of other processes.
>
> Accent is designed to support multiple instruction sets, so process state is defined in terms of the microcode interpreter. A process swap can occur at any point in the microcode execution designated as safe by the microcode implementor. On a process swap, Accent saves microcode registers; thus, a process swap can occur even in the middle of the execution of a macroinstruction. At present, microcode exists for Pascal and Lisp, and at least one project (DSN) has written application microcode to speed up their programs.

Communication between processes is through "messages" sent to and received from "ports", using a machine-independent interprocess communication facility, named "IPC." IPC is the "glue" that holds together a system of otherwise independent processes. In Accent, process interaction is always explicit and always takes place through IPC, thus it is a simple matter to put communicating processes on separate machines. In particular, since all access to the screen and keyboard is also made through IPC, you can operate and debug a distributed system from a single console. We have taken several programs consisting of multiple processes, debugged them on a single machine, and then distributed them with few surprises.

> IPC is an asynchronous message-passing system with buffering, flow control, and protection mechanisms. The kernel maintains structures called ports to which messages may be sent and from which messages may be received. Ports are referenced within a process by a local name. For convenience, the term port has come to mean the local name for a port as well as the kernel object. To send or receive a message, a process must have "send" or "receive" rights for a port. These port rights are managed by the kernel and represent a form of "capability". Like capabilities in other systems, port rights can be transferred within messages, and cannot be forged. This property provides the basis for a number of security mechanisms within the system.
>
> To make message passing efficient for large messages, Accent performs a "virtual copy" instead of moving data physically from one location to another. This is accomplished by manipulating address maps so that pages in two address spaces are mapped to the same physical page. The copy semantics are maintained, however, because any attempt to write into a shared page causes a page fault, at which time the page is physically copied.

2.1. Server Processes

The Spice operating system consists of a collection of server processes in addition to the Accent kernel. These servers are started automatically at system "boot" time. In this section, the function of each of the main server processes is summarized to give a global picture of the system structure.

Sapphire manages the display, keyboard, and graphics tablet and serves two main purposes. First, it provides facilities to support graphics and text IO. Second, it protects the screen from uncontrolled access by the servers or user programs. Without this protection, any program could write arbitrarily into portions of the screen ("windows") allocated to other programs. Operating system servers, as well as user programs, communicate with Sapphire via IPC, although simple text IO is often handled by the language implementations.

Sesame is the Spicc file system. Actually, Sesame consists of a collection of cooperating servers, which handle different aspects of the system: naming, file access, authorization and authentication, and migration. At present, only part of Sesame is implemented[2], and each PERQ is treated as a separate file system. Ultimately, Sesame will implement a global name space, spanning all the machines in the department, and allowing the protected sharing of files and named objects between various users. The system will also include automatic file migration to archival storage. At present, however, Sesame provides a simple form of network file access and limited protection.

The Message Server is a process that extends the local IPC facility over the network. When a message is sent to a port on a remote machine, it actually goes to the local network server, which then forwards it to the remote network server over the ethernet. The remote message server then sends the message as a local IPC message to its destination port. This whole process is transparent to the sender and receiver of the message in most respects.

The Spice message server also allows a process to post a name for a port. Another process can then present the same name and obtain the right to send messages to the port. The message server will use the network to search for remote ports if a local lookup fails. These functions are performed by a piece of the message server that is sometimes called the Message-Name Server; it is not really a separate server in the sense of being a separate process, although it is easier to think of the Message-Name server in these terms.

The Net Server provides packet-level access to the ethernet.

The Environment Manager provides mechanisms to define new commands, to set default pathnames and search lists, and to define variables. These facilities constitute the environment in which the shell, servers, and user programs execute.

Finally, we come to the Process Manager, a server that takes care of a number of tasks related to process

---

[2]The interim version of Sesame goes by the name of "Sesamoid" although many people ignore this distinction and stick to "Sesame". We will ignore the distinction and hope you will not be confused. If you hear Sesamoid, think Sesame and you won't be too far afield.

creation and process termination. The Process Manager is also used to allocate display windows on behalf of Sapphire.

2.2. Applications

Although Spice is a new system, several substantial applications are already available. In this section, we survey the application programs available in the current Spice release, but we omit any discussion of file utilities and compilers at this point.

- There are two Spice editors. The older, called **Oil**, is a screen editor in the style of EMACS. Oil is a simple editor, lacking the frills and extensibility of EMACS, but is otherwise quite comfortable and has been in use for a long time. A newer editor, **Hemlock** is a faithful implementation of EMACS.

- A drawing program, **DP**, is available to produce illustrations and technical drawings. DP has been used extensively for electronic circuit diagrams, and post processors are available to perform consistency checking and to generate wirelists from circuit diagrams. Illustrations generated by DP can be incorporated into documents produced by Mint, the Spice document preparation system.

- **Mint** is a powerful system for document formatting and layout. It incorporates multiple interpreters (currently a Scribe-like language, DP, and Plot), and will format output for either the PERQ screen or the Dover printer.

- **Update**, a version control and file transfer system, can be used to move sets of files between a PERQ and a Vax Unix. Update provides the basic mechanism for distribution and updating of Spice software. You will find it very useful for maintaining your own software and document files.

- **Mercury** is a mail system based on Rdmail. It is also available on Vax Unix under the name **Hg**.

- In the area of programming tools there is, of course Pascal, the original language for the PERQ and the implementation language for the Kernel and most of the servers, commands and utilities. In addition, **SubAda**, a compiler for a subset of Ada is available. SubAda is link-level compatible with the Pascal compiler and it can also be used on Vax Unix.

- Spice Lisp is just beginning to run on PERQs. We expect to see large performance improvements and some application programs in the months to come. One of the first application programs is the **Hemlock** editor.

- An IPC interface generator, called **Matchmaker**, a powerful debugger called **Kraut**, and a macro preprocessor, **Pasmac**, are also available. Matchmaker generates client/server interfaces that look like procedure calls. Most of the details of IPC message formats and system calls are thus hidden from the programmer. Kraut is a symbolic debugger for Pascal and SubAda programs with a conditional tracing facility and the ability to display values in user-defined formats. Pasmac is a macro preprocessor which has been used to generate Pascal and microcode programs.

- A C compiler and a Unix-like environment (**Spoonix**) are also beginning to run on PERQs; this will make a large body of existing programs available on the PERQ.

2.3. A Glimpse at the Future

Before describing the present system in greater detail, it is worth mentioning some of the directions in which Spice is moving. This is not a complete list, nor is it a promise to actually implement anything.

An algebraic programming environment for Ada, called Ada+, is also being developed. This will integrate several aspects of programming environments within a single, coherent implementation.

With new languages comes the problem of producing and maintaining multiple interfaces to servers. We are working on an interface specification language which is machine and language-independent and a new Matchmaker program to generate IPC interfaces for each language.

While Mint is currently a non-interactive document formatter, its internal structure was designed to support interactive document design and incremental formatting. We expect to see Mint evolve into an interactive system. In addition, there is interest in extending Mint to produce and format mathematical, Hindi, and musical notations.

The present operating system offers very little security. Work is in progress to make network communication more secure through encryption and the use of a secure central authentication server. Mechanisms to provide protected resource sharing are also being implemented.

Finally, a debugger for distributed, multiple language systems is being implemented. This debugger will be based on Kraut.

3. Getting Started - The Hardware

3.1. The Perq

Spice software, including its operating system Accent, currently runs on the Perq Systems Corporation PERQ personal computer. There are three types of PERQs at CMU, PERQ1, PERQ1A, and PERQ2. The main hardware features and the differences between the models are described in Appendix B.

On a not-too-detailed inspection, PERQs share the following features:

- high-resolution graphics using a 1024 x 768 bit-mapped raster display;
- optional 1-Mbyte (double-sided, double-density) floppy drive;
- internal (Winchester) disk;
- Ethernet interface;

- tablet and pointing device;
- standard keyboard with special function keys;

In addition, there are a few features of the PERQ which may not be immediately obvious on inspection but which the user should know about:.

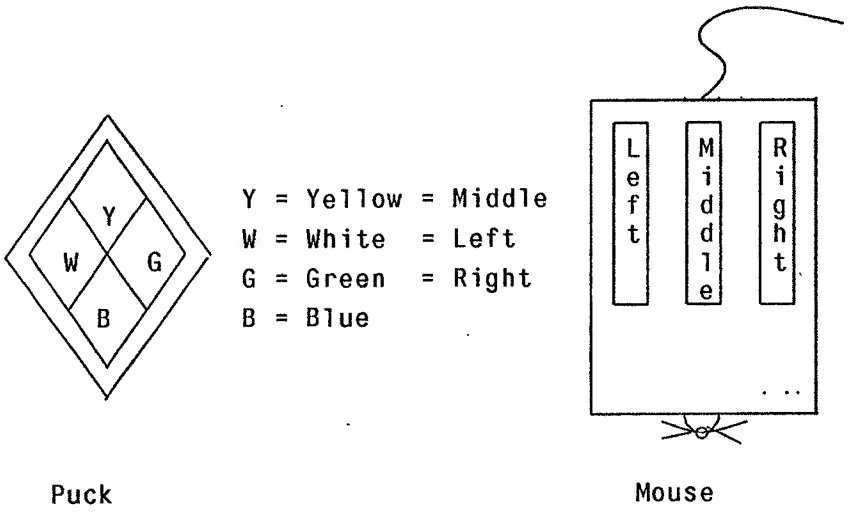
- The **Boot** button causes the PERQ to restart its microcode. On the PERQ1 and PERQ1A, the boot button is located at the rear of the keyboard, just above the cable. On the PERQ2, the boot button is in the bottom right side of the display cabinet, next to the keyboard cable plug.
- The **Diagnostic Display (DDS)** is a three digit display which is set by the microcode to help maintainers diagnose problems. On most PERQ1s and PERQ1As the DDS is located under the keyboard although in older models, it is hidden behind the front panel of the main processor cabinet. On the PERQ2, it is located in the center of the base of the display cabinet.
- The **Brightness** control for the screen is located on the back of the display cabinet.
- The **Volume** control for the audio output is also located at the back of the display cabinet.
- PERQ1s and PERQ1As come with an **Angle Adjustment** screw for adjusting the tilt of the display screen; it is located on the back of the display cabinet. PERQ2s must be adjusted by manually tilting the display cabinet into the desired position.
- There are two kinds of tablets. The larger ones have a **Tablet Reset** button, located on the right edge of the tablet, underneath the overhang.
- A **Power Connect** light, which indicates that power is connected to the PERQ, is located at the bottom rear of the main PERQ cabinet. It should be on whenever the PERQ is plugged in.
- The **Power On/Off** switch is located in the front panel of the main PERQ cabinet.

Section 3.3 on Booting Accent, explains how some of these features are used.

3.2. The Tablet and Pointing Device

PERQs are supplied with two kinds of tablet/pointing device combinations. The larger tablet comes with a four-button puck. The buttons on the puck are colored and arranged in a diamond (See figure below). The smaller tablet is supplied with a 3-button mouse. The buttons are arranged horizontally and are referred to as **left**, **middle**, and **right**. The left mouse button is equivalent to the white puck button; the middle mouse button is equivalent to the yellow puck button; the right mouse button is equivalent to the green puck button. There is no mouse button equivalent to the blue puck button. Although the puck and the mouse are quite different in appearance and in (internal) mode of operation, from now on we will use the term "mouse" to refer to either kind of pointing device.

The PERQ can only read the position of the mouse when it is on the tablet. That is, one may move the pointer



large distances (on the screen) using successive small "wipes" with the mouse (on the tablet). It should be lifted well clear of the tablet between wipes.

Presssing and releasing a button are two separate actions and different programs would do different things in response to such actions. In addition, their behavior might be affected by moving the mouse after pressing a button and before releasing it. Be sure you understand what is expected by the programs (when in doubt, experiment but save your files first!) Examples of these programs include Sapphire, the Shell, DP and, Oil.

3.3. Booting Accent

In this section we describe the normal procedure to be followed in turning the PERQ on and bringing up Accent. Most of the time the operation will proceed very smoothly and in two or three minutes you should be able to use the machine. However, if the operation does not proceed as scheduled, you might want to consult Appendix C for more information about the procedure, known problems, and corrective actions you might take.

Assuming that the PERQ is initially turned off, the following steps may be used to turn it on and bring up Accent:

- Turn on the power by pushing the power on/off switch on the front panel of the main cabinet. You should hear the sound of the fans.
- After a few minutes, during which the disk is coming up to full speed, the PERQ should

automatically "boot" itself. Some PERQs, however, do not seem to do so. In that case, push the small Boot button located at the rear of the keyboard (on PERQ2s, the boot button is located on the lower right hand side of the display cabinet). Many people regularly push the button after turning on the power.

- After the disk reaches full speed, some strange patterns will appear on the screen, lasting about 10 seconds. At this point the boot micro-code decides what operating system to boot by examining which keys on the keyboard are depressed:
    - Pressing none of the keys is equivalent to pressing (lower case) "**a**", which boots the standard Spice operating system, Accent.
    - Pressing the "**b**" key boots the old PERQ Operating System, POS, which is used at CMU as a backup system.
    - There may be some other key combinations for other "bootloads," but the average user need not be concerned with them.
- One of the first things Accent does after booting is to set its clock. It sends out a message on the Ethernet asking for the time. If it gets a response, it sets the clock and continues. If it gets no response (perhaps because the PERQ is not connected to the network), then it asks the user for the date and time roughly as follows (the exact wording might change in later releases of Accent),

```
Found paging partition sys:paging
Paging size = 10076

Accent System S4, 'a' boot
Starting Time Server... Done.
 Asking for time... no answer 3RCC
No time server accessible.
Enter time of day as DD-MMM-YY HH:MM:SS
```

  At this point the user should type an appropriate response, such as:

```
01-Nov-83 14:04:00
```
this is a 24 hour clock

  Typing just RETURN (or entering a date or time in a wrong format) will cause Accent to ask you again.

- Beware that, unless your PERQ is not connected to the Ethernet, this failure to read the date and time might be a symptom of a problem with your machine or the network. If it happens often, report the problem. In any event, be sure the time you type is correct (i.e. as correct as your wrist watch... don't be fuzzy). If you type in the wrong time or worse yet, the wrong date, you might have problems later one. If you realize that you made a mistake, it is better to end the session (Section 5.8) and then reboot.
- Accent will then start several server processes, and finally start Sapphire, the window manager program.

3.4. The Display

Once Sapphire is running, the first window to appear is a small "icons" window, at the bottom of the screen. It contains the Sapphire icons: small pictures of the Process Manager, the shell, and any windows created by the user. A few seconds later, a second window appears at the top of the screen. This is the Process Manager window and is used to display system information (e.g. the list of programs currently running).

> During the booting sequence, the Process Manager window traces the start of two additional auxiliary network servers (MsgServer and NetServer). If you are not connected to the Ethernet or if you do not have runnable (**.run**) files for these servers, the operation will fail and you will be notified through the Process Manager window. These servers are not critical for local operations; you can still use most of the Spice software. If they are not running, you will not have transparent access to the network, although programs like **CMUFTP** and **Update** may still work properly.

Finally, a large window appears in the middle of the screen and then a pointer, represented as an arrow appears towards the middle of the window. This is the current listener window. A shell (command interpreter) will start running on this window, "listening" to commands from the keyboard or mouse.

4. Getting Started - The Software

After Sapphire is running and has created the three initial windows and the pointer, the Spice shell (command interpreter) is ready to accept command lines and execute them. However, before learning about the shell, you must learn and become comfortable with a few conventions used to specify directories and file names.

Appendix D describes the notation conventions used throughout this manual. At this point you might want to read it and become familiar with the conventions.

4.1. Devices, Partitions, and Directories

Files in Spice are named much like those in Unix[3], using pathnames: a list of names terminating with a directory or file name. In Unix, elements in a pathname are separated with the character "/". In Spice, the separator is the character ">".

In contrast to most operating systems running on the time-shared facilities in the CS Department, currently, on a PERQ you must be aware of the physical devices used to store your files. The typical PERQ might have, in addition to the regular Winchester disk, a Floppy disk drive. Thus, when you specify a file name, you must start by indicating which device contains the file. In addition, the PERQ Winchester disk is usually divided into several chunks of storage, called "partitions". When you specify a file name stored in the disk, you must also specify the name of the partition containing the file.

---

[3] Unix is a Registered Trademark of Bell Laboratories

The name sys is reserved for the PERQ's Winchester disk. If you use the Mount command to mount a Floppy disk, the new device will have a different name (e.g. floppy or XYZ.)

Partitions are divided into directories and dircctorics can be further divided into (sub)directories, etc. to any number of levels. If you are familiar with the Unix file system, the organization of the Spice file system should look familiar. In particular, a partition behaves like a directory, and you can store files at the partition level, without having to go into the (optional) directories inside the partition.

Partitions play a special role in the PERQs. Operating system kernels and servers are stored at the partition level. During the boot sequence, the PERQ microcode looks at the key being pressed to select a version of the operating system. The device and partition names of these system files are stored in a fixed area of the disk (the disk "boot area") and that is how the microcode finds them. You can not just move (or rename) these files without letting the system know about it to update the disk boot area.

Having to specify a device name, followed by a partition name, followed by zero or more directory names, before you get to the actual file name is not very efficient. Sesame, the Spice file system, provides defaults for most of the components in a complete file name and you won't have to worry too much about device names and only a little about partition names. Examples throughout this chapter should make all of this clear.

4.2. PathNames

A complete file name or pathname for short, corresponds to a path in a "tree" of devices, partitions, directories and files. The root of the tree is always a device name. There are two main kinds of pathnames: absolute pathnames which start with a leading ">" and relative pathnames (i.e. relative to the current directory) which start with a name. Absolute pathnames are explained in Scction 4.2.4 while relative pathnames are explained in Section 4.2.6. In addition, sometimes you will find pathnames that start with a leading "<". These involve the concept of *LogicalNames*, explained in Section 4.2.2; they are a third kind of pathname.

Since a directory can contain both files and other (sub)directories, the Directory command lists the latter with trailing ">" characters to tell them apart.

Sesame follows the Unix convention of using the reserved name ".." for the parent directory. For example, the pathname >A>B>C>D>..>..>E is equivalent to >A>B>E.

4.2.1. File Names and Extensions

A filename in Spice is a sequence of letters, digits, underscores ("_"), periods ("."), hyphens ("–"), and dollars ("$"). An extension is the portion that appears at the end of a filename, following the last period (if there is no period, the filcname has no extension; this is often refered to as having the "null" extension). In Sesame there is nothing special about extensions, although certain conventions have been established (e.g. **.cmd** is used for command filename extensions). These conventions are described in Appcndix E.

In Sesame, a complete filename also includes a "version number" as part of the name. The version number of a file is specified by a ";" followed by number, after the file name,

```
>Sys>User>foo.bar;1
```

This first release of Spice software does not make use of the version numbers.

No special significance is attached to the characters in a filename or even to the capitalization of the letters. The latter is an important detail to keep in mind if you are used to the Unix conventions or if you intend to move files between your PERQ and a Vax Unix.

Name your files using whatever mnemonic system you prefer. However, whenever possible, do follow the conventions for file extensions. This will make your life easier and will prevent obscure bugs arising from programs that make (sometimes) unwarranted assumptions about your file names.

4.2.2. Logical Names

The syntax <*LogicalName*> at the beginning of a pathname is used as a synonym for another, perhaps longer, pathname. That is, a *LogicalName* can be thought of as being replaced by another string of characters which is its expansion. This is useful for abbreviating pathnames or to hide details about precisely which device or partition is to be used. For example, the logical name **<boot>** corresponds to the partition from which Accent was booted. If Accent was booted from **>Sys>Spice>** (i.e. device Sys, partition **Spice**), then the name **<boot>LibPascal>Sapphire.pas** is equivalent to **>Sys>Spice>LibPascal>Sapphire.pas**.

At present, the following logical names are defined:

<current> the user's current working directory, as set by the **Path** command (Section 6.2.)

<default> the user's default search list for files, as set by the **SetSearch** command (Section 6.2.) Search lists are explained in Section 4.2.3. Note that the command **SetSearch** keeps **<current>** at the head of **<default>**. That is, by default the system looks first in the current working directory before looking somewhere else.

<run> the system's default search list for runnable (.run) and command (.cmd) files. Search lists are explained in Section 4.2.3. **run** is initially (at boot time) set to <current>,<boot>. That means that the file system will look first in the current working directory before looking in the boot partition.

<boot> the partition from which Accent was booted. Normally, this is **Spice** but can be changed by the user: typing alternative boot selection characters on the keyboard at boot time results in the booting of different operating systems (e.g. Accent or POS) or operating system versions (e.g. Accent S3 or Accent S4), and these could be stored in different partitions.

<dev> the device from which Accent was booted. At present, this is the PERQ's local Winchester disk, whose device name is Sys (POS can also be booted from floppy

disks; someday we might be able to do this for Accent. Another future possibility is to boot across the network, from a disk associated with another PERQ).

You can define new logical names using the **Define** command, and you can examine the list of currently defined logical names using the **Show** command (Section 6.2).

For example, the **<run>** search list can be (re)defined by the **Define** command, as in

```
Def <run>  <default>,>Sys>User>Utilities
```

in which case, the system looks first in the **<default>** search list before looking in the Utilities directory in the User partition. It will not look anywhere else for runnable **.run**) or command **(.cmd)** files.

4.2.3. Search Lists

A "search list" is a convenient way of specifying lists libraries (directories) containing files and programs of general use. There are two predefined search lists, known by the logical names **<default>** and **<run>** (See Section 4.2.2.)

The **<default>** and **<run>** search lists differ in the way their contents are specified: directories can be inserted into or removed from **<default>** via the **SetSearch** command (Section 6.2), while any search list, including **<run>**, can be set via the **Define** command (Section 6.2.) **SetSearch** exists mostly for compatibility with **POS**.

Pascal programmers will want to add the library **<Boot>LibPascal** to their **<default>** search list. **LibPascal** contains source files describing the interface to most servers and system utilities. These interfaces are created by MatchMaker, thus allowing Pascal programs to send and receive IPC messages with a minimal amount of trouble. **LibPascal** is added to the search list by typing one of the following command lines to the shell:

```
SetSearch <Boot>LibPascal
Define <default> <Boot>LibPascal/After=1
```

Searching for files is done automatically by the file system. That is, the system will always look through a list of alternative directories in the proper search list. The user can control the scope of the search by adding or removing directory names from the search list. The most restrictive case would be to have an "empty" search list. Sesame will then look only in the current working directory

4.2.4. Absolute PathNames

An absolute pathname begins with the character ">" and is used to specify a file or directory name, independently of the current working directory. Absolute pathnames have the following syntax:

>DeviceName>PartitionName{>name}

That is, a device name, followed by a partition name, followed by 0 or more directory names (the last one could be a filename).

4.2.5. Network PathNames

Besides Sys, the name for the user's own PERQ, the device name in an absolute pathname could be the name of another user's PERQ. This provides a simple naming scheme for accessing files and executing programs stored elsewhere on the Ethernet. However, a few minor requirements must be satisfied for this to work.

- The other machine must be on, running Accent.
- The other user must allow external (network) access to her files.

  At present, a user can allow either full network access (including the right to delete), read-only access, or no access at all (not even the right to look at a filename). The default is read-only access. Future releases of the system will provide more sophisticated access control options.

- You must know the other user's PERQ, partition, directory, and file names. This facility is quite new and no conventions have been established; try asking the other user.

The name by which your PERQ is known throughout the network is defined in a file named SysName in the **<boot>** partition (i.e. **<boot>SysName**). You are free to edit this file and write any name you wish in the first line of the file (the first time you use the PERQ, the file might not exist; you must create it with Oil or Hemlock.) Common sense would seem to suggest that you use your last name (if this is truly a personal machine), an office number (if this machine is shared with officemates), or even a project name. Do not use "HarryQBovick" or "Sys" for obvious reasons.

The name stored in SysName is registered in the Network when you boot the machine. If you just changed it, the new name will not be known until you re-boot.

4.2.6. Relative PathNames

As explained before, a relative pathname (i.e. a pathname that begins with a *name* instead of a ">") can be interpreted in two ways, depending on the context in which it is used. Thus, which file is referred to in a relative pathname is command or program dependent.

In the first interpretation, a relative pathname is considered to be prefixed by the logical name **<current>**. Thus, a pathname like **A>B** is equivalent to **<current>A>B** which in turn is equivalent to the absolute pathname **>sys>User>Gumpertz>A>B** if the current directory is **>Sys>User>Gumpertz>**.

In the second interpretation, a relative pathname is considered to be prefixed by the logical name **<default>**. Thus, a pathname like **A>B** is equivalent to **<default>A>B**. Directories in a search list are searched until the first match is found. If the search list **<default>** included several pathnames (say, **>Sys>Larry>**, **>Sys>Curly>**, and **>Sys>Moe>**), then **A>B** stands for one of the following filenames **>Sys>Larry>A>B**, **>Sys>Curly>A>B**, and **>Sys>Moe>A>B**, depending of the order of the directories in **<default>**.

The directories in the <default> search list are searched in order; once one or more file names are found in a directory, the search terminates. Thus you might not get a complete search of all the alternatives in all the directories in this search list.

A command like **delete** assumes the first convention (this is a good precaution: if you make a mistake when typing the file name, the system won't accidentally delete a file in a different directory.) The Oil editor assumes the latter (editing the "wrong" file causes no harm, you can always leave the editor without altering the file.) Since **<default>** includes **<current>** as the first place to search, the second convention will look first for a match in the current working directory before trying for matches in other directories.

Programs like the linker, would use the the second convention to find program modules but the output, a **.run** file, will appear in the current directory, even if all the modules came from a different directory.

4.2.7. PathName Restrictions

For the time being, only directories, files, and ports can be specified with a pathname. In the future, the same naming convention will be extended to include many other types of objects.

The total length of any pathname must not exceed 255 characters although individual components of a pathname can be up to 80 characters long.

Because upper- and lower-case letters are considered distinct by the Unix file system, some care with capitalization may be necessary when dealing with Unix. The simplest way to avoid problems is to use only lower-case letters when naming a file that might be stored on a Vax Unix system.

4.2.8. Wildcards

Special "wildcards" may be used when specifying pathnames to some (but not all) commands. This allows one to name several different files at one time. Typing a "*****" as part of a pathname, matches 0 or more characters in the place where the "*****" appears. For instance, if a directory contains files **abc1, abc12, abc2,** wildcards could be used as follows:

ab*1	matches **abc1**; a one-character match.
abc*	matches all three files.
abc1*	matches **abc1** and **abc12.**
***bc*2**	matches **abc12** and **abc2.**

The complete set of wildcards is a bit more elaborate. It includes * (matches 0 or more characters), & (matches 1 or more characters), % (matches exactly 1 character), 'a or 'A (matches any letter), '0 (matches any digit), and '@ (matches any non-alphanumeric character.) Any of the these patterns can be quoted (so that it matches only itself) by preceding it with an apostrophe (').

In the near future, this pattern matching is going to be substantially simplified. In particular, only two patterns will be available: * (matches 0 or more characters) and % (matches exactly 1 character.)

With the exception of the Directory command, wildcards may appear only in the last component of a pathname:

```
Delete A>B>*                                    is legal
Delete A>*>C                                    is not legal
Directory A>B>*                                 is legal
Directory A>*>C                                 is legal
```

For a more detailed discussion of the file system, see *Sesame: the Spice File System.*

5. Getting Started - The Shell

The Spice shell is a fairly simple command interpreter. It prompts the user with a black-filled "❯" and reads and executes command lines, one at a time.

Because the various commands were implemented over a period of time with no explicit agreement upon conventions, users will encounter a number of different styles for interacting with programs. With time, we hope that some consistency will emerge. Initial results of one effort, the Cooperative User Interface (Cousin) project, will soon be released. This, and other standardizations, are strongly encouraged. Nevertheless, there have arisen a few patterns of usage style that should be mentioned.

Appendix D describes the notation conventions used throughout this manual. Before reading this chapter, you should be familiar with the conventions.

5.1. Command Line Conventions

The standard command line consists of the name of a command, followed by any number of "input" arguments separated by commas, followed by any number of "output" arguments, followed by "switches". Switches, which are used to specify desired variations in command behavior, usually start with a "/", and can take a value, as shown below,

```
Edit MyTextFile.mss                          Invokes the Oil editor on a user's file
SetSearch <boot>LibPascal,>sys>user>w        Adds two directories to the current search list
Directory *.Pas ~x                           Lists the names of all files with extension .pas into a file named x
Delete Test*/NoConfirm                       Deletes all files beginning with test, without asking for confirmation
Link Module1,Module2,Module3 ~ProgramName/Map
                                             Links several modules into one runnable program and produces a link map
link test/StackSize=1024/Map                 Specification of a stack size
```

That is, input arguments are separated by commas, as are output arguments. The two classes are separated from each other by either spaces and/or a tilde; the latter is preferred. Spaces are permitted after, but not instead of, the commas separating input arguments.

Formally, the command line syntax is as follows

Command[*SwitchList*] [*InputArgList*] [~][*OutputArgList*] [*SwitchList*]

where both the *InputArgList* and *OutputArgList* are of the form:

Argument[*SwitchList*]{,*Argument*[*SwitchList*]}

and a *SwitchList* is of the form:

{/*Switch*[=*Value*]}

See Appendix D for a description of the conventions used to represent the syntax of commands.

Most commands normally accept simple names as arguments. Sometimes, it may be necessary to specify an argument which includes characters that would be interpreted as delimiters (such as space, comma, or slash). To allow this, there is a "standard" (and unusual) quoting convention: precede the character with an apostrophe ('). For example,

```
alias clean delete' *.oil',*$'/noc
```

represents a string that includes an space, a comma, and a slash. This convention is quite different from the Fortran, PL/I, or Pascal quoting convention of enclosing the entire string in quotes. Furthermore, it makes no provision for specifying the zero-length (null) string. Note that many commands do not fully support this convention anyway.

Most Spice commands and utilities either ignore the difference between upper- and lower-case characters or follow the rule that the original capitalization should be used when displaying a name but any combination of cases should be accepted when matching one name against another. For example, assume that we use the Oil editor to create a new file:

```
Edit MyTextFile.mss
```

After we finish editing the new file, its name will have the capitalization used above although the shell will accept the following alternatives:

```
Type mytextfile.mss
Type MYTEXTFILE.MSS
Type mYtEXTfILE.MSS
```

In each case, the newly created file be typed on the screen. The **directory** command always spells filenames using the original capitalization.

What constitutes a valid argument or switch is program- or command-dependent and you must consult the proper documents. Over time, a few conventions have arisen. For example, many commands would respond to the **help** switch by typing a short description of their operation, and the correct syntax to invoke them, as in:

```
directory/help
```

Notice that the switch is spelled out "**h**" "**e**" "**l**" "**p**" and should not be confused with the keyboard key labelled HELP. The latter is not used by many programs.

Some programs, like Chat, use the HELP key as a "help" command, to distinguish it from the word **help** being typed to

another host machine. Chat commands are specified through combinations of special keyboard keys since it has no way of distinguishing between a command to Chat and a command with the same spelling but meant for the other machine.

5.2. Special Command Lines

If the command line starts with an "@", then the rest of the command line should be the name of a command (**.cmd**) file from which shell commands may be read. Command files may be nested arbitrarily but there is currently no facility for passing parameters to command files. Note that only lines containing shell commands, not lines read by application programs, may appear in a command file.

If the command line starts with an "!", then the remainder of the line is treated as a comment and is ignored. This is especially useful to document command files.

If the command line has an "&" appended, then the command will be executed in another process which runs in parallel with the shell and uses another window. The process manager will prompt the user to indicate the size and position of that window. This suffix is ignored if the command line begins with an "@". To run a command file in parallel, first create another window and then invoke the command file in that window's shell.

The character "%" at the end of a command line turns on a loader debugging switch.

If the command line has a "^" appended, then the command will be executed with the debugger enabled. That is, the process will be suspended before it starts running and the Kraut debugger will be invoked. If the command line has both a "^" and an "&" appended (in that order!), then both of the conventions above will be followed. The "^" suffix is ignored if the command line begins with an "@".

Redirection commands similar to those in Unix may be used: "->" is used to redirect output, "<-" is used to redirect input, and "|" is used to pipeline one program's output into another program's input.

A double semi-colon ";;" may be used for sequential process execution.

5.3. Using the Spice Shell

The shell looks for a command name (an arbitrary string of characters delimited by a space, "=", or "/") at the beginning of the command line. The remainder of the line is saved for the program implementing the command. If the shell can find a runnable (**.run**) file with the same name as the command, then that file is executed. Otherwise, the shell checks whether the command name matches one of its built-in commands.

Remember that .run files are looked up in the **<run>** search list, not in the **<default>** search list. If you are not happy with having different search lists, you can always define **<run>** to be equal to **<default>** by including the following line in your profile (**<boot>InitialShell.cmd**) file:

```
def <run> <default>/Global
```

A list of the built-in commands may be obtained using the "?" command; individual commands are documented in the Spice Commands and Utilities Manual. Built-in command names can be abbreviated, provided there is no ambiguity with another built-in command. Thus, for instance, the **SetSearch** command could be typed as **sets**, the **Directory** command as **dir**, etc.

The shell supplies a default file name to several commands. For example, if you want to compile a program immediately after editing it, you can omit the file name from the Compile command. Commands that currently use or set this default file name are Edit, Compile, Link, and Run. In addition, the Type command uses this name as a default parameter but will not change the default.

5.4. The Intra-line Editor

The shell provides a facility to edit command lines as they are being entered, before typing RETURN. By default, the editing commands are modelled after the EMACS editor although the actions associated with the keys can be modified by the user (See Appendix H.)

5.4.1. Command Line Editing Functions

The functions understood by the intra-line editor (assuming the EMACS-like defaults) are given below. Note that capitalization is important.

CTRL a — Move cursor left to the beginning of the line.

CTRL b — Move cursor left (back) one character.

CTRL B — Move cursor left (back) one word.

CTRL d — Delete the character to the right of the cursor.

CTRL D — Delete the word to the right of the cursor.

CTRL e — Move cursor right to the end of the line.

CTRL f — Move cursor right (forward) one character.

CTRL F — Move cursor right (forward) one word.

BACKSPACE *or* CTRL h *or* DEL
Delete the character to the left of the cursor.

CTRL H — Delete the word to the left of the cursor.

TAB *or* CTRL i *or* CTRL I
Insert spaces up to the next tab stop (spaced every eight characters from the left margin).

CTRL k — Delete characters to the right of the cursor, all the way to the end of the line.

CTRL n — Retrieves (from an internal command buffer) a previously typed command line. See Section 5.4.3, below for further details.

CTRL p — Retrieves (from an internal command buffer) a previously typed command line. See Section 5.4.3, below for further details.

CTRL t	Transpose the two characters to the left of the cursor.
CTRL v	Displays the following screenful from the transcript buffer. See Section 5.5.
CTRL V	Displays the previous screenful from the transcript buffer. See Section 5.5.
CTRL w	Turns "wrap" mode on. That is, output lines that would be too wide for the current window width are continued onto the next line, i.e. are "wrapped" around. The wrapping is not permanent; if the window width is changed, the lines are redisplayed as per the new width. See also CTRL W, below.
CTRL W	Turns "wrap" mode off. That is, output lines that are too wide for the current window width are truncated. This is not a permanent truncation; if the window witdh is changed, the lines arc redisplayed as per the new width. See also CTRL w, above.
OOPS *or* CTRL u	Delete characters to the left of the cursor, all the way to the beginning of the line.
CTRL LF	Enables "more-mode" to control scrolling. See Section 5.4.4, below. More-mode is disabled by default.
CTRL \	Disables "more-mode" to control scrolling. See Section 5.4.4, below. More-mode is disabled by default.
LF	Allow scrolling to continue. See Section 5.4.4, below. The command is ignored if output is not suspended by CTRL LF.
INS *or* ESC	Completes a partially specified filename. See Section 5.4.2, below for further details.
CTRL ?	List the file names that could match a partially specified filename. See Section 5.4.2, below for further details.

5.4.2. Automatic FileName Completion

The INS (ESC on PERQ2s) kcy is used to ask the shell to complete a filcname which has bcen partially typed by the user. The shell examines the characters to the left of the cursor and searches for a filename that begins with that sequence. If a unique filename exists, the shell inserts the rest of the filename and moves the cursor to the right of the last character, as if the user had just typed the entire filename.

If the cursor is in the middle of a name, it is as if the "wildcard" character "*" had been expanded. For instance, if the cursor is under the "a" in the middle of the filename "foar" when INS is typed, then "foar" will be expanded to "foo.bar" provided, of course, that "foo.bar" is the only file name that could apply (see below.)

If no filcname with the given prefix exists or if the prefix is ambiguous (i.e. more than one filename matches

the prefix), the screen "flashes" for an instant, to indicate a failure to complete the filename; type one or more characters before trying INS again, until the shell reacts by completing the filename.

If in doubt about the possible candidate file names, type CTRL ? to get a quick listing of the current candidates. CTRL ? typed all by itself, at the beginning of a command line, will list all files in the current directory[4]. Typed after a partially specified filename, CTRL ? will list all filenames that could complete it (if no candidate filenames are found, the screen will flash, to indicate a search failure).

5.4.3. Retrieving Previous Command Lines

The CTRL p and CTRL n commands are used to retrieve previous command lines. Previously typed command lines are kept in a "ring" and the user can move around the ring, typing CTRL p to move backwards or CTRL n to move forwards. The old command line appears on the screen, with the cursor positioned to the right, as if the user had just typed it in.

Successive CTRL p or CTRL n commands can be typed until the desired line is retrieved. It can then be edited, if necessary. Typing RETURN terminates the editing and the command line is then interpreted by the shell. Notice that the cursor does not have to be at the end of the line in order to type RETURN

5.4.4. Scroll Control a.k.a. "more-mode"

The shell has the ability to suspend output to a window if it would cause information to scroll off the top of that window. This is analogous to the TERMINAL PAUSE (ON) END-OF-PAGE feature in the TOPS-20 system, or "more-mode" in the Vax Unix systems.

To enable this facility, type CTRL LF, as a shell command, at any time during the session. From then on, whenever the shell notices that output is about to scroll off the top of the window, it stops and displays the last line in the window in reverse video (white letters over black background). Typing LF allows output to continue.

> Typing anything other than a Sapphire CTRL DEL command, will cause scrolling to continue. However, whatever was typed is saved and used as input the next time the shell or the program doing the typing wants to read something from the keyboard.
>
> Do not get into the habit of typing RETURN when you mean LF. How would you like to see the following prompt from a program, right after you resume scrolling with RETURN?
>
> ```
> Delete ALL your files [Yes]?:
> ```

To disable the scroll control facility, type CTRL \ as a shell command. This is the default mode after booting Accent.

---

[4] If the current directory is empty, this command displays all the filenames in the first non-empty directory in the default search list.

If you forgot to set "more-mode" and some important program output scrolled off the window, you might still be able to read some of it by retrieving the previous transcript buffer(s). See Section 5.5, below.

5.5. Session Transcript Buffers

The shell keeps a transcript of the session (i.e. user command lines and program output that appeared on a window) in an internal buffer. One can peruse this buffer by typing CTRL V and CTRL v to move backwards and forwards, one screenful at a time.

The buffer keeps about three windowfuls of text, thus the size of the window determines how much is retained in the buffer. Changing the size of the window, alters the size of the buffer for the remainder of the session (or until the next window size change).

The transcript buffer does not behave like the command line buffers mentioned in Section 5.4.3. It can not be used to "replay" a portion of session or to retrieve previous commands. Typing anything, while displaying some previous portion of the transcript, simply advances to the end of the transcript and inserts the user input at the place were the cursor was positioned before the user started redisplaying the session transcript.

5.6. Creating Additional Windows

There are several ways to create, move, and reshape windows. Sapphire provides complete sets of keyboard commands, pop-up menus, and mouse button commands to perform these and other operations. For detailed information on the creation and use of windows, see *User's Guide to the Sapphire Window Manager.* In this section we only describe the use of the Shell command.

To create an additional window, type

```
shell
```

A blinking cursor in the form of an upper-left-corner will appear on the screen. Position the cursor where you want the upper left corner of the new window. Press and release any button. The cursor will change to a lower-right-corner with an arrow. Move the cursor to where you want the lower right corner of the new window. Press and release any button.

In order to communicate with the new window, you have to designate it as the current listener. To do this, move the cursor into the window and press any button. When the button is released, the border changes from black to gray to indicate that the newly selected window is now the current listener.

The current listener does not change by simply moving the cursor to another window, as was the case in Canvas, the original Spice window manager. If one forgets to press and release a button, any characters (including commands) typed are sent to the last window designated as listener. *Caveat emptor.*

The process of creating a new shell from an existing shell is often referred to as "spawning". Spawned shells inherit a copy of the "environment" of the shell from which they were created. This includes things such as the definitions of **<Current>** (the current working directory) and **<Default>** (the current search list).

5.7. Initial Command Files and User Profiles

The Spice shell provides for user customization via command files. After booting, the shell looks for a file with the name **InitialShell.Cmd** in the **<boot>** partition and then executes the commands contained in that file. Typically, this command file would define new commands and environment variables, set server priorities, and define logical names.

> Creating a new window with the pop-up menu "shell" command or the keyboard CTRL DEL s command, invokes the execution of two command files: **<boot>ShellProfile.Cmd** and **<boot>ShellCommands.Cmd** on the new shell, if these command files exist, and in that order. The new shell inherits the environment of the initial shell.
>
> Beware that creating a new window with the **shell** command does not invoke any automatic execution of command files.

5.8. Ending a session

There is no "logout" command in the Spice shell. To terminate a session type the command **Trap** to the shell. This command invokes the Accent kernel debugger; wait until the debugger prompts with the character ">" for a command and then turn your PERQ off. If you change your mind, and decide not to terminate the session, you can quit the kernel debugger and return to the Shell by typing the command "q" to the debugger.

> The kernel debugger is a rather sinister looking character; it shares Darth Vader's taste in clothing, and announces its presence by writing over a blacked out portion of the screen, disregarding window boundaries. Grown-ups have been known to cry over the debugger's blood curling battle cry: "106", "106"..... Sometimes it will cry "144", thus tricking you into believing that it is actually beginning to like you and that it might even recover..... fear not, eventually you will get your "106".

6. Basic Commands and Application Programs

Appendix D describes the notation conventions used throughout this manual.

6.1. On-line Help

The help command provides explanatory text related to a word provided as an argument by the user:

```
Help word
```

The convention used is to look for a file with the name **<default>***word*.**help** and type it on the screen if it is found. For instance,

```
Help kraut
```

is equivalent to

```
Type <default>kraut.help
```

In addition, some commands and application programs recognize the **help** switch. If the help command describes above does not provide the needed information, try the following:

name/**help**

The "?" command lists the built-in shell commands followed by the user defined commands.

6.2. File and Environment Manipulation Commands

6.2.1. Directory

This command lists all file names in the current directory. To obtain a list of **.run** files, type

```
dir *.run
```

File names in other directories can be obtained by specifying the name of the directory, including wildcards, if necessary:

```
dir <UserUtilities>RunFiles>*.run
```

The shell intra-line editiong command CTRL ? , when typed at the beginning of a command line, can be used to get a quick listing of the current directory. It is faster than Dir but does not have the complete functionality of the regular command: you can not specify switches, pathnames, wildcards, etc. If the current directory is empty, this command lists the contents of <boot>.

6.2.2. Path

This command changes the current directory. If you type

```
path
```

and the current directory is >Sys>Spice the following message will appear on the screen:

```
Old path = >Sys>Spice>
New default path: (CR to exit)
```

If you do not want to change the current directory, type RETURN. If you to want to change to a different directory, for example, ..>new, type

```
..>new>
```

You can type the new path name in the same line with the command, as in

```
path ..>new>
```

in which case you will not be prompted for a confirmation. "Path x" is equivalent to "Define <current> x", where "x" is a valid pathname.

6.2.3. MakeDirectory

This command is used to create new directories:

```
MakeDir NewDirectoryName
```

For instance, to create a (sub)directory named **foo** inside the current working directory, type

```
MakeDir foo
```

6.2.4. Copy

This command is used to make copies of files:

```
Copy OldFileName NewFileName
```

For instance, to create a copy of an existing file, say, **bar**, in a new file named **>sys>user>more**, type

```
copy bar >sys>user>more
copy >Rashid2>Spice>User>Petal.pas MyPetal.Pas
```

The last example shows how to obtain a copy of a file from another PERQ on the EtherNet (i.e. from device **Rashid2**, partition **Spice**, directory **User**, file **Petal.pas**)

The *OldFileName* can have wild cards in it as long as the *NewFileName* has the same wild cards, in the same order. All files that match the source will be copied by taking the characters that match each wild card and putting these characters in the corresponding place in the destination.

6.2.5. Rename

This command is used to change the name of an existing file:

```
Rename OldFileName NewFileName
```

For instance, to change the name of **bar** to **more**, type

```
rename bar more
```

the old file **bar** does not exist anymore.

Renaming only works if the two filenames are in the same partition. To achieve the effect of **Rename** across partition boundaries, you would need to do a **copy** to the destination partition (OldFileName -> NewFileName), followed by a **delete** in the original partition.

6.2.6. Delete

This command deletes files. To delete a file say, **test.pas** from the current directory, type

```
delete SomeFileName
```

Delete irrevocably destroys the specified file (there is no undelete command). If you use wildcard characters in the specification of a file to be deleted, the command will prompt you for confirmation on each possible file.

6.2.7. Show

This command displays the environment variables and their values. See the Define and Alias commands, below.

6.2.8. Define

This command is used to define (or redefine) a variable in the environment:

`Define` *VariableName* *VariableValue*

where *VariableName* is of the form **foo** or **<bar>** (the latter defines a logical name which can then be substituted in pathnames).

To remove a variable from the environment, type

`Define` *VariableName*

that is, define it with the "empty" value.

The *VariableValue* can be followed by the switches **/local** or **/global**. The **/local** switch makes the variable visible only to this shell and to shells spawned from it (creation or "spawning" of new shells is explained in section 5.6). The **/global** switch makes the variable visible to all shells. The default is **/local**.

6.2.9. Alias

This command is used to define new commands in the environment, e.g.,

`Alias` *CommandName* *CommandValue*

where *CommandValue* is some previously defined command or program name (including arguments or switches). Typing *CommandName* as a command is equivalent to typing *CommandValue*

This substitution does not include the RETURN normally issued at the end of a command line, thus one could type the following

CommandName *FurtherArgumentsOrSwitches* RETURN

as a command line.

6.2.10. SetSearch

To add pathnames to the <default> search list, type

```
SetSearch PathName,...,PathName
```

SetSearch treats the default search list like a stack: the last pathname added goes to the head of the list and is the first directory to try in a search. If instead of a pathname, you type a hyphen ("–") the pathname at the head of the list is removed from the search list. Alternatively, you can type just the command name:

```
SetSearch
```

The shell displays the current contents of the search list and prompts the user for input. If a pathname is typed, it will be added to the front of the default search list, if a "–" is typed, the first name in the list is deleted. Typing RETURN causes the program to exit with no further changes.

6.3. Producing Documents

Spice provides tools for creating, editing, formatting, and printing documents and drawings.

6.3.1. Oil

To invoke Oil, one of the Spice editors, type

```
Edit SomeFileName
```

Oil commands are similar in style to those in Emacs. For more information see *Oil: The Spice Ascii Editor.*

6.3.2. Mint

To invoke Mint, the Spice document formatter, type

```
Mint
```

Mint will prompt you for the information it needs. Mint formats documents for the Dover printer or the PERQ screen. Mint commands are similar to those in Scribe. For more information see *User Manual for Mint: The Spice Document Preparation System.*

6.3.3. DP

To invoke DP, the Spice drawing program, type

```
DP
```

DP allows users to create illustrations and circuit diagrams. DP can produce output for the Dover printer directly or in conjunction with Mint. For more information see *DP: The Spice Drawing Program.*

To modify an already existing illustration, you can type

```
DP SomeFileName
```

6.3.4. Dover

The Dover program sends documents created on the PERQ to the Dover for printing. It is invoked with

```
Dover
```

There are many switches available to enable you to specify where to ship files and how to format them. Type **Dover/help** for a complete list of switches.

You can also provide the name of the file to be printed in the same command line,

```
Dover SomeFileName
```

6.4. Using the Ethernet

PERQs at CMU are connected to the Ethernet. You may use your PERQ as a terminal to connect to another machine on the Ethernet, to send and receive mail, and to transfer files.

6.4.1. Chat

The **Chat** program allows you to use your PERQ as a terminal connected to another Ethernet machine. **Chat** simulates an HDS Concept terminal. If you type just the name of the program,

```
Chat
```

You will be prompted for the remote host name. When you have established a connection with the remote host, simply log in as you would from any terminal. Type the HELP key for help and a list of commands (e.g. how to quit **chat**).

If you type a host name in the same command line,

```
Chat e
```

Chat will not prompt you for a host name and will connect you to the "e" host.

6.4.2. Cmuftp

The **Cmuftp** program is used for general purpose file transfers between your PERQ and any host machine on the EtherNet. **Cmuftp** will prompt you for additional commands with the character ">". Type **help** for a list of commands and further explanation.

6.4.3. Update

The **Update** program is used to transfer files between the PERQs and the Vax systems. **Update** is the main vehicle for distributing Spice software and it has some built-in knowledge about the CMU-CS-Spice and CMU-CS-CFS Vaxen (these are the machines where Spice software is developed and stored). Of course, you can use this program and any other Vax Unix system on the Ethernet to maintain you own software and documents. Update is described in *Update: The Spice File Transfer Facility.*

6.4.4. Mercury

Mercury is the Spice mail program. Its commands are similar to those in RdMail on the TOPS-10 systems or Hg on the Vax Unix systems. For more information, see the Mercury User's Guide supplied with this manual.

Listing Mercury as an ethernet "user" is not quite correct. Mercury proper only deals with your local mailbox; it allows you to read mail messages and to write messages to be mailed. Mercury relies on an auxiliary server (MailMan) to do the actual transmission of messages over the Ethernet. MailMan normally runs in the background and its operation should be transparent to most users.

7. What To Do When Something Goes Wrong?

7.1. Reporting Problems

The primary error-reporting mechanism is mail sent to "Spice@CMU-CS-Spice" describing the problem.

For some serious problems, there is specific information that you can collect and can be useful when trying to diagnose the problem:

7.1.1. Failure to Boot

If the PERQ does not boot properly, please report the number displayed in the Diagnostic Display (DDS) (normally located on the underside of the keyboard on PERQ1s and PERQ1As; it is located on the front pannel of the screen cabinet on PERQ2s).

7.1.2. Accent Crashes

An Accent kernel crash is normally indicated by entering the kernel debugger which types out white characters on a black background, ignoring Sapphire windows.

Together with an error number, the kernel debugger sometimes will write a short message identifying the source of the problem. One problem you might find is described as "Running out of paging space". This happens when the paging partition used by Accent to support the virtual memory is exhausted; this could be due to having allocated too little space to the partition or after many hours of work, when many process and file pages have been "paged out". Rebooting the system will normally solve the problem. Other, more serious problems, are reported as "Illegal Memory Reference" (error 106) or "Uncaught Exception" (error 144).

The kernel debugger has a primitive command language. Sometimes the system is able to recover from the crash and typing the command q will let you continue interacting with the shell. If the condition persists, your only recourse is rebooting.

It would be very helpful to the maintainers if you could report the message printed on entering the debugger plus the information printed out by the commands "c", "s", and "f". For the "f" command, the first few lines of output are sufficient.

7.1.3. Failure to Run

Sometimes, when trying to run a previously linked program (i.e. a **run** file) you might see error messages of the form:

```
ALoad: Incorrect length for run file >sys>spice>LibPascalInit.RUN;1
Shell: Error in spawning >sys>user>guest>CALC.Run;1 Code is 124
```

Where the command you tried to execute (i.e. the program you tried to run) was **calc.run**. This error occurs whenever the program and **LibPascalInit.Run** (a system program normally linked with most Pascal programs) are inconsistent. In other words the program was not linked with the current **LibPascalInit.Run**

This is not a serious problem (all that happens is that the program fails to be loaded and executed; thus the two error messages). To solve the problem you must relink the **run** file (i.e. **calc.run** in the example). If the **seg** (object code) files produced by the Pascal compiler are around (e.g. this is one of your own programs), just issue the same **Link** command that you used the first time you linked the program and you will have a compatible **run** file.

If you do not have the **seg** files around, which is common if this is a library program or if you got it from some other user, then the **link** command to use is:

```
link/force libpascalinit.run, calc.run/main/include
```

This error is likely to occur if you brought to your PERQ (say, via **update**) a new version of **LibPascalInit.Run** and you already had **Calc.Run** or if you brought a (new) version of **Calc.Run** that was linked with a different version of **LibPascalInit.Run**

Sometimes, if this problem happens too often with programs you retrieve over the network, you might want to get (via **update** a new version of **LibPascalInit.Run**; chances are you have an obsolete version.

7.2. Getting Out of the Kraut Debugger

Kraut is the name of the debugging program. Kraut will be invoked whenever an error is detected by the system (by the Process Manager, to be precise). If the error occurred in some system program it is unlikely that you will be able to do much debugging. You probably want to get out of Kraut, and back into the shell.

When the Process Manager invokes Kraut, the Process Manager window will change to reverse video (white letters on black background). A message stating the nature of the error (e.g., "uncaught exception") will appear, and the Sapphire cursor will prompt for the creation of a window (See Section 5.6). Once Kraut is

running in the window that you have just created, it will display some information regarding the status of the program when the problem occurred. This is information of interest to programmers and maintainers. After this is done, Kraut will prompt for debugging commands with the character "|".

To exit Kraut, make the new window the listener (move the cursor to the window and press and then release any mouse button; the border should change from black to gray) and then type the command **Quit**.

> If you want to report the bug to somebody (e.g. the program maintainer), type the command **Calls**, followed by the command **Report**, followed by the command **Quit**. Kraut will ask you for the mailing address of the intended recipient, your name, and whether you want to provide an additional message or not. If you decide to provide this extra information, Kraut will read lines of text, until you type a line that consists of just a period (".") in the first column. At this point, Kraut will package all the information and mail it to the specified destination, before terminating the debugging session.

You should now make the original window the listener; do this before you type any shell commands.

For on-line help about Kraut, type the command line **Help Kraut** to the shell, or type the HELP key while still in the debugger window. For more information see *User Manual for Kraut: The Spice Interim Debugger*

7.3. Partition and Scavenger

These two programs are used to fix problems with the disk. Both of them were written for **POS**, the original operating system for the PERQ and, until something similar is developed for Accent, you will have to boot the **POS** operating system (by using the **b** character during the boot sequence) in order to use them.

Scavenger is rather harmless and is used to recover the "free list" of a partition. The free list is a mechanism used by the file system to keep track of the free space in each partition (except for the **paging** partition, which is never used to store files and therefore does not have a free list). Sometimes, a system error will cause the free list to be clobbered. This causes no end of grief to the file system (it can not create or delete files; often it can not even look at filenames inside a directory). Normally, you do not want to do anything with that partition: things can get really ugly otherwise. When the file system suggests that you "scavenge" a partition, do it.

On occasion, Accent will write on the screen a one line message of the form "Scavenging 3745623000". The message would appear in reverse video (white letters on black background), towards the top of the screen, ignoring window boundaries. This is not the message "suggesting" that you scavenge a partition. The latter will appear in normal video and will follow some file system action (e.g. after deleting a file, when leaving the editor, etc.).

To run Scavenger, boot POS and type (to the POS shell),

```
Scavenger
```

Scavenger will prompt for some parameters. Until you become an expert, follow its lead and use the defaults it suggests. Finally, the program will ask you for the name of the partition to be scavenged. Type the partition name and let the program do its thing (the screen pyrotechnics are rather nice to watch.) After reading the disk blocks allocated to the faulty partition, Scavenger will try to reconstruct the files and the free list. If it finds some anomaly (e.g. a "broken" file) it will tell you. Sometimes it will ask you for information (e.g. a "file name") or confirmation (e.g. to fix an error in a "bad" file). Do what it suggests or ask an expert for help.

> Files are stored in the PERQ's disk as double-linked lists of blocks (512 bytes per block). An extra block is allocated as a "file header" and contains information about dates of creation and access, the name of the parent directory, the file length, etc. Although the usual reason for using this program is to recover the free list, the program is also useful to fix other problems, such as rebuilding split files and reconstructing directories that were accidentally deleted (the files are still there but not accessible).

Partition is a heavy duty "fix-it-upper". Although it provides some functions that you think you would like to invoke (e.g. splitting or renaming a partition), it can also do more amusing things, like deleting the whole disk in just as many keystrokes. Do not use it until you know what you are doing.

7.4. When Things Look Hopeless

Reboot.

However, to be safe, you should do so at some point when there is no danger of stopping the system while some disk input/output operation is in progress. If the shell is still capable of prompting for and executing commands, the safest way is to type the command Trap to the shell. This command invokes the Accent kernel debugger; wait until the debugger prompts with the character ">" for a command and then reboot.

If you can not type shell commands, then it is a good idea to wait for a period of time (something in the order of a minute), until no lights on the right-hand side, at the top of the screen, are flickering. This ensures that there is no disk activity. It is then safe to reboot.

Appendix A
SpiceTalk Glossary

This Appendix describes some terms commonly used when describing Spice facilities.

Term	Description
Address space	Instructions in a program refer to data and other instructions through their memory addresses. The address space of a program is the set of memory addresses that can be read from or stored into. See Virtual Memory.
Boot, Booting	In Spice, this word is used in two ways. First, there is the action of "booting" or starting Accent and associated servers after the machine is turned on. On some PERQs, booting does not happen automatically after power is turned on: the "boot" button must be pressed and released to get the machine going. The second use of the word is in **<boot>**, the logical name for the initial partition i.e. the partition acting as the current directory right after booting the machine.
Current directory	See Directory.
Current listener	The current listener is the window/shell combination that is "listening to" (i.e. reading input from) the user's keyboard input. The window is identified by having a thick, gray border instead of the normal thin, black border.
Cursor	Cursors are the small pictures used to "point" or select items on the screen. Typical cursors have simple shapes like arrows, underbars, etc. The underbar used by the intra-line editor to mark the current input position is also called a cursor.
Device name	In the Sesame file naming convention, filenames are specified as a sequence of (nested) directories names. The top level "directory name" (i.e. the first element in the sequence) is the name of the device where the file is stored. The device name can refer to a local device (e.g. Sys, the name for the local disk, inside the PERQ main cabinet) or to a device somewhere else on the network (e.g. **Rashid** could be the name of the local disk in the machine owned by user **Rashid**)
Directory	Directories provide a convenient way to group related files. There could be many reasons to put a file in a given directory (e.g., program files related to one project, manuscript files for a book, utility programs used in several projects, etc.) Directories can be nested, and a sequence of directory names, prefixed with a device name and a partition name, provides a unique path to a file.
Environment	The set of logical names, search lists, user defined commands and variables, etc. available to the shell (and user programs).
Free List	The Sesame file system keeps track of the disk space available to store files. The space is organized in blocks, 512 bytes long each. The free list is simply the list of all the currently available (i.e. free) blocks of storage that Sesame can use for a new file. Files themselves are also organized as lists of blocks. After removing a block from the free list, the newly allocated block can be written into and then linked with the file's list of blocks.

Icon — Icons are small pictures representing all of the existing windows, including those that are covered by other windows. Icons are displayed in the icons window, at the bottom of the screen. The icon corresponding to the current listener has a thick, gray border instead of the normal thin, black border.

IPC, Inter-Process Communication — The mechanism used to communicate between processes. Processes send or receive messages via ports which act as protected mailboxes.

Kernel, Kernel Call — A low-level machine operation, used to request Accent to do something. For instance, to send a newly created message, to allocate more memory to a process, etc. Kernel calls are similar to "supervisor" or "monitor" calls in other operating systems.

Logical name — A logical name is just an abbreviation used to shorten a complete pathname or to hide details such as device or partition names.

Message, Message passing — Messages are the pieces of information (e.g. data, commands, etc) passed between processes. See IPC.

More-mode — The ability to suspend output being sent to a window after sufficient lines has been printed to fill the window. This gives the user a chance to read it before it goes away. After reading the information, the user can resume output for another windowful.

Mouse — The mouse or puck is the device that slides on top of the tablet. Moving the mouse or pressing and releasing its buttons are meaningful actions to Sapphire, the servers, and user programs. The position of the mouse over the tablet it often associated with the position of the cursor on the screen. Thus, moving the mouse on the tablet usually changes the position of the cursor on the screen.

Path name — A path name is a complete specification of a file name, starting with a device name (which could be somewhere else on the network), followed by a partition name, followed by zero or more directory names, followed by the actual file name.

Paging partition — See Partition.

Partition — The local PERQ disks are divided into several large chunks of storage called partitions. In a pathname, the name of the partition containing the file appears after the name of the device, but before any directories inside the partition. There is a utility program under POS, called **Partition**, which is used to initialize, split, rename, or merge partitions.

paging is a special partition set aside in the local disk to help Accent implement the virtual memory of the servers and user processes. It is never used as, as the normal partitions, to store files.

Spice is the default name for the partition containing all the Accent and server files.

	The logical name **<boot>** stands for the actual partition name (you could be booting an experimental system, residing in a different partition in which case **<boot>** would be something other than **Spice)**
Port	Accent's version of a mailbox. Processes send messages to and receive messages from other processes via ports allocated and protected by Accent. See IPC.
Process	A process is the basic computation unit under Accent. Any number of processes can be running simultaneously, but their address spaces are separate to avoid interference between them. Cooperating processes communicate via messages and ports.
Puck	See Mouse.
Scavenging	The operation of examining the disk, checking the integrity of the free list and the files. Often used to recover from file system errors that cause the free list to be clobbered or a file to be truncated.
Scroll control	See More-mode.
Search List	An ordered collection of directory names, used by the file system to look for files when they are not found in the current working directory.
Server	Servers are the processes that complement the facilities provided by Accent to make up the Spice Operating System. Servers are usually started by Accent when the machine is booted and continue running in the background for the complete session.
Shell	A shell is a command interpreter program usually associated with a window, although not all windows need to have a shell. The shell also reads and edits (through the intra-line editing functions) the user's keyboard input before passing it along to the application program running on the window. The **Shell** command can be used to create a new window and to associate a shell with it.
Spice partition	See partition.
Sys	The device name for the local disk. It is the first component of a complete pathname within one PERQ. To access files across the network, the "device name" to use is the name of the other machine. This "network name", the name by which a PERQ is known throughout the network, is kept in a file **(<boot>SysName)** in each machine.
Tablet	The tablet is the device which, in conjunction with the mouse. provides a mechanism to move the screen cursor around, and to select (i.e. point to) objects, windows, etc. on the screen.
Tracker	A process that keeps track (hence the name) of the position of the mouse on the tablet by moving some cursor or icon on the screen.
Typescript	A process that reads characters from the keyboard on behalf of the Shell and other programs. A Typescript process uses key translation files to interpret keyboard keys,

some of which might be passed directly to the program, others could be translated into some other characters before passing them to the program, and yet others could be used by Typescript as "commands" not to be passed to the program. The intra-line editing functions of the Shell are implemented by a Typescript process.

Virtual memory — A technique by which the address space of a program can be made to appear much larger than the actual, physical memory in the computer. It is normally achieved by using disk storage as an extension to the physical memory. Under Accent, the paging partition is used to support the virtual memory of the server and user processes.

Window — A portion of the screen, usually associated with a shell or command interpreter. Any number of overlapping windows can be created but only one of them can be the current listener.

Appendix B
Perq Hardware Configurations

There are three types of PERQs at CMU, PERQ1, PERQ1A, and PERQ2. PERQ1As, the most common PERQ type at CMU, have the following hardware features:

- 16-bit micro-programmed processor;
- 16K 48-bit words of writable control store;
- high-resolution graphics using a 1024 x 768 bit-mapped raster display;
- 1-Mbyte of main memory;
- 24-Mbyte Winchester-technology disk drive;
- optional 1- (double-sided, double-density) floppy drive;
- 3-Mbps Ethernet interface;
- pointing tablet;
- standard keyboard with special function keys;
- RS232 & GPIB I/O interfaces;
- audio (speech) output.

The PERQ1 has 4K words of writable control store and a 3-Mbps Ethernet interface. The PERQ2 has a 40-Mbyte disk and two RS232 I/O interfaces.

Power consumption is roughly 700 W or 7A (it is a non-resistive load); this is comparable to a home air conditioner. It may, therefore, be operated on a standard 15A branch circuit. Under current Duquesne Light Company electric rates, the cost of using a PERQ at home is about $0.07/hour (ignoring slightly reduced home heating costs in winter and increased cooling costs in summer). Until availability, maintenance, and other issues are resolved, however, home use is not being encouraged. It has been reported that many offices at CMU share 20A branch circuits; using more than three PERQs at a time on such a circuit can trip the associated circuit-breaker. *Caveat emptor!*

Appendix C
Booting Accent

This appendix describes the booting sequenee in full detail. Under normal conditions, you ean safely ignore this material. Sometimes however, things do not seem to work as indicated in seetion 3.3. This material should be useful to identify the problem.

If the PERQ does not boot properly (the Diagnostic Display does not reaeh "400") and the material in this Appendix does not help to identify and eorreet the problem, please report the number displayed in the Diagnostic Display (DDS) (normally loeated on the underside of the keyboard on PERQ1s and PERQ1As; it is located on the front panel of the screen cabinet on PERQ2s).

Assuming that the PERQ is initially turned off, the following steps may be used to turn it on and bring up Aceent:

Power On

Turn on the power by pushing the power on/off switch in the groove on the front panel of the main eabinet. The fans should start; if you do not hear them, cheek to see that the machine is plugged in. If so, see if the small neon light in the lower right hand side of the back of the eabinet is glowing. If the light is out (indicating lack of power on the line), eheck the electrical outlet. If there is power at the outlet or the light is glowing and you still do not hear the fans, send mail to Spice@CMU-CS-Spiee, specifying the symptoms and the location of the PERQ.

Boot Button

After a few minutes, during which the disk is coming up to full speed, the PERQ should automatically "boot" itself. Some PERQs, however, do not seem to do so. In that case, push the small red Boot button loeated at the rear of the keyboard (on PERQ2s, this is a black button on the lower right hand side of the screen cabinet). Many people regularly push the button after turning on the power.

Diagnostic Micro-program

While the maehine is booting, a series of very regular patterns produeed by the diagnostie mieroprogram called VFY will appear on the sereen. They look like a set of binary trees, somewhat eomb-like.

Operating System Selection

Just after the diagnostic microprogram finishes (the patterns disappear and you get a blank screen, with some horizontal retracing lines on it), the boot microcode decides what operating system to use by examining which key on the keyboard is pressed. Pressing none of the keyboard keys is equivalent to pressing (lower ease) **"a"**, whieh boots the standard Spice operating system, Aecent. Other operating system/key eombinations might be available on your PERQ (e.g., POS is normally asociated with the "b" key.)

Because it may be hard to get the timing just right, a good method is to push the key down *before* pushing the boot button and then hold it (them) depressed until the operating system has announced itself.

> The boot selection key is read after the VFY has terminated and the DDS display shows "151" so in principle, you can wait until that point in the booting sequence. Some users have learned to recognize the VFY screen patterns and wait until the right time before pressing the boot selection key. Until you get the hang of it, follow the instructions in the last paragraph.
>
> Life is not that easy however: pressing the boot selection key for too long has been know to screw up many PERQ2s. If nothing seems to happen and the DDS display shows "152" you have one of these mutants. You will have to learn to release the boot selection key a couple of seconds after you see "151" come up on the DDS display.

Setting the Clock

After VFY has completed, the screen will become entirely white (with some bright, rolling, nearly-horizontal retrace lines crossing it) until Accent starts using the display.

One of the first things Accent does after booting is to set its clock. It sends out a message on the Ethernet asking for the time. If it gets a response, it sets the clock and continues. If it gets no response (perhaps because the PERQ is not connected to the network), then it asks the user for the date and time, in the following format:

```
01-Nov-83 14:04:00
```

Notice the format: this is a 24 hour clock

Typing just RETURN will just cause Accent to ask you again.

Starting the Servers

Accent will then start several "server" processes, and finally start Sapphire, the window manager program. Sapphire creates two windows on the screen, one near the top and one near the bottom. If this does not happen, or if the message "**GPIB?**" appears on the screen, it is possible that you tablet is not working properly. It definitively is not working if when you move the mouse, the cursor on the screen does not move.

If you have a PERQ with a large tablet and a puck, there are several things worth checking, each of which has caused problems in the past:

- The puck was not placed on top of the tablet during the boot sequence.
- The tablet's power supply, a large black box, might not be plugged in.
- The cable from the power supply to the tablet may not be firmly plugged in to the back of the tablet. Note that this plug is polarized, so you need not worry about plugging it in upside down.
- Either end of the GPIB cable from the PERQ to the rear of the tablet might not be firmly connected (with its hold-down screws tight).
- The puck may not be plugged in to the front of the tablet.

- The tablet might require reinitialization, which can be done by pressing the tablet-reset button on its right edge. In some extremes cases, it might be necessary to turn the PERQ off and then on again.

If your perq comes with a small tablet and a mouse, it could be the case that the tablet cord is not plugged into the display cabinet.

After correcting any of these problems, it might be necessary to reboot.

Appendix D
Notation Conventions

We have attempted to use the following notations throughout this manual when describing the syntax of commands:

SYMBOL	MEANING
|	used to separate alternatives
( )	used to indicate the scope of alternatives for | where it would not be otherwise obvious.
[ ]	used to enclose an optional feature, something that can appear 0 or 1 times.
{ }	used to enclose something that can appear 0 or more times
Bold	a literal, something that must be typed as it spells.
Italic	a metaname, something that stands for a group or class of names. For instance, a file name can be described as: *Name.Ext* that is, a *Name*, followed by the literal ".", followed by an *Ext*. *Name* stands for <u>any</u> filename (i.e. a sequence of leters, digits, underscores, periods, and hyphens), not just "**N**" "**a**" "**m**" "**e**".
CTRL	used to prefix a control key (i.e. a key that must be typed while the keyboard "control" key is down). For instance, CTRL u indicates typing the 'control' and 'u' keys together. The keyboard control key acts like a conventional typewriter shift key. Nothing happens when it is typed or pressed by itself and you can keep it down while typing several CTRL keys in a row.
ESC, INS	the "escape" key (marked as ins on PERQ1s and PERQ1As, and as acc ese on PERQ2s).
DEL	the "delete" key (marked as **rej del** on PERQ2s).
HELP	the "help" key (not the word "**h**" "**e**" "**l**" "**p**").
LF	the line-feed key
RETURN	the carriage return key

Because the documentation has been gathered from several distinct sources, we may not have been as successful as desired. Please report any discrepancies by mail to Spice@CMU-CS-Spice.

Appendix E
Filename Extensions

An extension is the portion of a filename that appears at the end, following the last period (“.”). In Sesame, the Spice file system, there is nothing special about extensions. However, certain conventions are used in the system and these are described in this Appendix.

Backup files conventionally have a “$” as the last character of their name; they take the form *Name.Ext*$. Some programs (e.g. **Oil**) make a copy of a file into a a backup version, with the same file name and extension (with the “$” at the end) before modifying a user's file. **Oil** also creates transcript files containing the **Oil** commands used while editing a file. These transcript files have “**.oil**” as a suffix to the original filename.

A list of the standard extensions and how they are used follows.

.ada	This extension is used for SubAda source programs.
.bin	Micro-code object files. These files are produced by the micro-placer program, PrqPlace.
.boot, .mboot	The operating system and microcode interpreter files.
.c	C source files.
.ckp	Temporary transcript files created by **DP**.
.cpp,.i	C preprocessed (intermediate) files (during compilation).
.cmd	A number of programs accept commands from a file as well as from the keyboard. Files that contain the commands usually have this extension. See also and **.kmd**, below.
.dfs	These files are used to share definitions between Pascal and microcode programs.
.defs	These files contain definitions of server interfaces that are input to Matchmaker.
.dp	**DP** drawing files.
.DR	In the interim version of Sesame, directories are implemented as files. These files appear in a directory listing with a trailing “>” but are internally represented as having the extension .DR. Only users of POS should run into this extension; if it shows up anywhere else please send mail describing the situation to Spice@CMU-CS-Spice.
.exe	C executable files.

.h	C include files.
.help	Files containing help about a subsystem use this extension.
.lib	C shared library files.
.lmd	C linker command files.
.keytran, .ktext	Sapphire uses **.keytran** files to determine the mapping of keys on the keyboard and buttons on the mouse to "commands" for the servers, utilities, and user programs. **.ktext** files are the "source" files used by the KeyTran compiler program **KeyTranCom** and **.keytran** files are the "object" files produced by **KeyTranCom.**
.kmd	Kraut command files.
.kst	Character set definitions for use with the PERQ display screen are kept in files that have the extension **.kst.**
.micro	Microcode source files. These files are used as input to the micro-assembler, PrqMic.
.mint, .mss	Manuscript files, suitable for input to the document formatter, Mint, have this extension.
.o	C object files.
.oil	Backup file for oil. It contains the editing commands used the last time the file was edited with **Oil.** It can be used to replay the editing session in case the system failed to save the user's file. Not to be confused with the "**$**-terminated" file name which contains a copy of the user's file.
.pas	Pascal source files.
.pasmac, .pm	PasMac source files. Because filenames under Vax Unix must be fairly short, the extension **.pm** is sometimes used instead on that system.
.press, .prt	Formatted documents, suitable for printing on the Dover, have one of these extensions. The latter extension is used only for "temporary" press files that can be recreated easily and so can be deleted without much worry.
.rel, .rsym, .int	These three file types are temporary files that are produced by the micro-assembler and the micro-placer programs. They can be deleted after a **.bin** file has been created.
.run	Runnable files produced by the Linker have this extension.
.s	C assembly code files.

.seg	Object code files produced by the Pascal and SubAda compilers. **.seg** files are used as input to the Linker and contain the code that will be executed when the program is run.
.slisp, .sfasl	Spice list source code and compiled code files, respectively.
.sym, .qmap	Files with these two extensions are created by the SubAda and Pascal compilers to pass information to the **Kraut** debugger. The **.sym** file contains the names of the variables and routines declared in the program; the **.qmap** file contains a mapping between Q-code offsets in the **.seg** file and corresponding source statements (in the **.ada** and **.pas** files).

Appendix F
Predefined File Names

In this Appendix we describe a few files whose names are known to some programs (e.g. the Shell) and thus, have predefined meanings. Do not create files with these names for any purpose other than what is described below.

<boot>SysName This file consists of a single line of text containing the "name" by which your machine will be known throughout the network. You are free to edit this file and write any name you wish in the first line of the file (the first time you use the PERQ, the file might not exist; you must create it with your favorite editor.) Common sense would seem to suggest that you use your last name (if this is truly a personal machine), an office number (if this machine is shared with officemates), or even a project name. Do not use "HarryQBovick" or "Sys" for obvious reasons.

Do not edit this file without first checking with all the other users of the machine; they might count on it having a specific name. Never change the name of a public machine in the Spice rack.

The name stored in SysName is registered in the Network when you boot the machine. If you just changed it, the new name will not be known until you re-boot.

<boot>InitialShell.cmd

The Spice shell provides for user customization via command files. After booting, the shell looks for this file and then executes the commands contained in that file. Typically, this command file would define new commands and environment variables, set server priorities, and define logical names.

<current>ShellProfile and <current>ShellCommands

These files are used to provide initial Shell commands, to be executed when a Shell is being created by Sapphire.

To be precise, Shells created by Sapphire (typing CTRL DEL followed by s) inherit global environment variables and execute ShellProfile and ShellCommands (in that order) if these files exist. On the other hand, Shells created by the Shell command inherit the environment of the parent shell but do not execute any command files.

Appendix G
Son et Lumiere

Accent does not provide much by way of audio entertainment, the visual amenities are however, worth a trip to the nearest operating PERQ.

At the top of the screen, above the Process Manager window, there are a number of small (about 1x5 mm) rectangles. They are called "lights" and are used by Accent to indicate the state of the processor. If all lights are off (white), the system is idle. The meaning of each light, counting from the left, is as follows:

1. (unused)
2. DIRTY (currently rather meaningless; may be removed)
3. IDLE
4. MEMORY CLEANUP (no free pages; looking for unmapped pages)
5. CREATE PHYSICAL SEGMENT
6. NOT ENOUGH ROOM FOR PHYSICAL SEGMENT (error condition; failure in 5)
7. FIND A FREE VP RECORD (same as 4 for VP pages)
8. ALLOCATE A KERNEL AST BLOCK (probably unused)
9. (unused)
10. MSGS WAITING NON-NIL (File I/O in progress)
11. (unused)
12. (unused)
13. FIND LEAST USED PAGE (when 4 doesn't find any free page)
14. (unused)
15. WRITE FAULT (Probably disabled)
16. READ FAULT (Probably disabled)
17. FINDINTREE (Probably disabled)
18. PROCESS PAGE (Probably disabled)
19. (unused)
20. (unused)
21. CREATE SHADOW SEGMENT
22. (unused)
23. DISK READ
24. DISK WRITE
25. (unused)
26. (unused)

Appendix H
Key Translation and the Intra-line Editor

Some Spice programs such as DP, the Shell, Sapphire, and Oil bind certain character sequences typed from the keyboard to certain commands. This is accomplished by *key translation files.* A key translation file is a mapping of keys or key sequences typed into the keyboard to commands accepted by an application program. If you look for files with the extension .keytran on the <boot> partition, you will find various key translation files: The file **oil.keytran** defines the keybindings for the OIL editor, **dp.keytran** defines the keybindings for the DP drawing program, **chat.keytran** defines the keybindings for the CHAT virtual terminal program, and the file **<boot>ts.keytran** defines the commands understood by the intra-line editor in the shell.

This appendix shows, by example, how to define a (new) key translation file for the intra-line editor. It is by no means an introduction of how to create general key translation mappings or how they work internally. You must become a **Sapphire** expert in order to understand all the details.

To define a new key translation file you have to start with a .ktext file. The .ktext file contains a description of the key translations in textual format, whereas the .keytran file contains them in a binary representation.

A .ktext file consists of two parts: a command section and a keytranslation section. The command section contains a list of constant (number) declarations defining all the commands that can be bound to keys. Of course, they have to be known by the application, you cannot just make up your own constants. For example, the typescript process might equate the command **cFDEL** with the numeric value **54**: **cFDEL** deletes the character next to the current cursor position. To make the command known in the key translation file, insert

```
cFDEL = 54      ! Delete next character.
```

in the command section of the .ktext file. Exclamation marks are regarded as the beginning of a comment. Anything following a "!" up to the end of a line is regarded as comment.

The keytranslation section binds keyboard keys or key sequences to the commands defined in the command section. Keys are in general denoted by their character prefixed by a quote (') or the string printed on the keyboard. Note that characters are case sensitive. For example, 'a denotes the key lower case a and 'H the upper case H. **BACKSPACE** denotes the BACKSPACE key on the keyboard. The CTRL key is denoted by **CONTROL** and the space bar by **SPACE.**

A simple key binding is

```
CONTROL 'd        = cFDEL
```

which binds the key sequence CTRL d to the command **cFDEL.**

It is possible to bind sequences of keystrokes to commands. For example,

```
CONTROL 'a + CONTROL 'b + CONTROL RETURN = cFDEL
```

is a legal key binding: It binds the sequence CTRL a CTRL b CTRL RETURN to the command **cFDEL.**

The following file contains all commands known to the typescript process and redefines them to a **FLASH** style instead of the default **EMACS** style[5].

In the example, observe that the command section is prefixed by the words **DEFINITIONS** and **COMMAND**, the key translation section is prefixed by the word **KEYTRANSLATIONS.** The file terminates with the word **END.**

```
! Key translation file for the Typescript: Flash version
DEFINITIONS

COMMAND
        PosResponse     = 40! Don't do anyting.
        cFIRST          = 50! Go to beginning of line.
        cLAST           = 51! Go to end of line.
        cNEXT           = 52! Go to to next character.
        cPREV           = 53! Go to previous character.
        cFDEL           = 54! Delete next character.
        cBSP            = 55! Delete previous character.
        cEREOL          = 56! Delete to end of line.
        cERBOL          = 57! Delete to beginning of line.
        cTAB            = 58! Insert a TAB.
        cTRANSP         = 59! Twiddle the two before the current cursor.
        cRETURN         = 60! Insert a RETURN after the current cursor position.
        cREDISP         = 61! Redisplay the current listener window.
        cDELNWD         = 62! Delete the next word.
        cDELPWD         = 63! Delete the previous word.
        cNEXTWD         = 64! Move cursor back one word.
        cPREVWD         = 65! Move cursor forward one word.
        cPREVPAG        = 66! Retrieve last command from command buffer ring.
        cNEXTPAG        = 67! Retrieve next command from command buffer ring.
        cUNBLOCK        = 68! ??
        cSETMORE        = 69! Set current listener window into MORE mode.
        cDELMORE        = 70! Set current listener window into SCROLL mode.
        cUP             = 71! Scroll up one page.
        cDOWN           = 72! Scroll down one page.
        cLISTPATH       = 73! List all files in current directory.
        cCOMPPATH       = 74! Complete the file name or type all files with current prefix.
        cNOOP           = 75! Don't do anything.
        cILLEGAL        = 76! Illegal command.
        cBELL           = 77! Ring the Bell.
```

[5]FLASH is a text editor on Xerox's Alto personal computer. Some users prefer it over the standard Emacs-like shell intra-line editor.

Appendix I
Spice Project Contributors

Mike Accetta	David Ackley	David Adam	David B. Anderson
Daniel A. Aronson			
Nora Bailey	J. Eugene Ball	Mario R. Barbacci	Robert V. Baron
James Bolc	Lars Borg	Bernd Bruegge	Jacob Butcher
Amy Butler	Beth Byers		
Nick Caruso	Monica Cellio	Bill Chiles	Richard Cohn
Ellen Colwell			
Roger B. Dannenberg	David Dill	Marc Donner	Daniel Duchamp
Ivor Durham			
Carl Ebeling	Edith Eligator	Jeff Eppinger	
Scott E. Fahlman	Kimberlee Faught	Neal Feinberg	Kathleen Ferraro
Charles Fineman	Robert Fitzgerald	Len Ford	
David Garlan	Joe Ginder	Dario Giuse	David Golub
Richard H. Gumpertz			
A. Nico Habermann	Samuel P. Harbison	Greg Harris	Phil Hayes
Steve Henderson	William D. Herndon	Cynthia Hibbard	Peter G. Hibbard
Paul N. Hilfinger	Steve Hoffmann	David Hornig	Michael L. Horowitz
Susan Jennings	Sandeep Johar	Jeff Jones	Michael B. Jones
Gail E. Kaiser			
Jim Large	Gregg Lebovitz	Eric Lehmann	Richard Lerner
Paul McAvinney	Don McCracken	David B. McDonald	Rob MacLachlan
Bill H. Maddox	Horst Mauersberg	Brian Milnes	Brad Myers
D. David Nason	Thomas D. Newton	David Nichols	Joe Newcomer
Allen Newell			
Roy Ogawa			
Douglas W. Philips	Kathryn Porsche		
Richard F. Rashid	Raj Reddy	John Renner	George R. Robertson
Walter van Roggen	Steve Romig		
Robert D. Sansom	Mahadev Satyanarayanan		James B. Saxe
Don Scelza	Sharon Schanzer	Dana Scott	Nancy A. Serviou
Samuel E. Shipman	Lee Shumacher	Marc Singer	Knut Skog
Edward T. Smith	Alfred Z. Spector	Guy L. Steele Jr.	Suzanne Stevenson
Robert G. Stockton	Pedro Szekely		
Jeff Taylor	Mary Thompson	Mark Tucker	
Jon A. Webb	Bruce Whittaker	Skef Wholey	Francis C. Wimberly
Keith Wright			
Michael Young			
Dean Zarras	Barbara J. Zayas	Edward R. Zayas	John Zoll
Leonard N. Zubkoff			

```
KEYTRANSLATIONS

        Control 'k      = PosResponse ! Make CTRL K a NOOP
        CONTROL 'h      = cFIRST
        CONTROL ';      = cLAST
        CONTROL 'l      = cNEXT
        CONTROL 'j      = cPREV
        CONTROL 'd      = cFDEL
        BACKSPACE       = cBSP
        CONTROL 'e      = cEREOL
        CONTROL 'E      = cERBOL
        TAB             = cTAB
        CONTROL 't      = cTRANSP
        RETURN          = cRETURN
        CONTROL '#      = cREDISP
        CONTROL 'w      = cDELNWD
        CONTROL 'W      = cDELPWD
        CONTROL 'L      = cNEXTWD
        CONTROL 'J      = cPREVWD
        CONTROL 'n      = cNEXTPAG
        CONTROL 'y      = cPREVPAG
        LF              = cUNBLOCK
        CONTROL LF      = cSETMORE
        CONTROL '\      = cDELMORE
        OOPS            = cERBOL
        CONTROL 'i      = cUP
        CONTROL ',      = cDOWN
        CONTROL '?      = cLISTPATH
        INS             = cCOMPPATH
        CONTROL 'g      = cBELL
END
```

After you have created a file **tsflash.ktext** with the definitions above you want to create the corresponding **.keytran** file. This is done by invoking the keytran compiler **keytrancom** which compiles the **tsflash.ktext** file into a binary **tsflash.keytran** file.

If there are any errors in the **.ktext** file, the compiler suppresses the generation of the **.keytran** file and writes error messages into the window and into a file with the extension **.ERR**. The keytran file will be created only if there are no serious errors. This is shown in the following dialog (User input is underlined):

```
keytrancom ts.flash.ktext
Key-Translation Compiler V2.1

File with Keytran definitions (Ktext File): tsflash.ktext
File for binary output [tsflash.keytran]: tsflash.KEYTRAN
File for Pascal output [TsFlashKDefs.PAS]: TsFlashKDefs.PAS
File for errors [TsFlash.ERR]: TsFlashKDefs.ERR

* WARNING -- Command cNOOP defined but not used
* WARNING -- Command cILLEGAL defined but not used
Writing Defs file: tsflashKDefs.PAS
Writing Keytran file: tsflash.KEYTRAN
There are 28 entries which take 106 words.
Maximum hash chain length is 4 and there are 14 hash conflicts.
** Reported 2 warnings.
** These are all listed in file tsflash.ERR
```

To test the new keytran file, copy it into **<boot>ts.keytran** file. The intra-line editor commands of any new shell created afterwards will be bound according to the new key sequences.